WHAT IS CLAIMED IS:

Sub A5

~~1. A computer implemented method for selecting a code image during a reboot routine, comprising:~~

~~maintaining multiple code images in a memory device;~~

~~executing a first operation routine;~~

~~incrementing a first counter if the first operation routine succeeds;~~

~~executing a second operation routine;~~

~~incrementing a second counter if the second operation routine succeeds; and~~

~~using the first and second counters to select one of the code images from the memory device to execute.~~

~~2. The method of claim 1, further comprising:~~

~~designating one code image as non-operational if the first counter is a first value and the second counter is a second value, wherein one other code image not designated as non-operational is selected to execute.~~

~~3. The method of claim 2, wherein the first value is greater than zero and the second value is zero.~~

~~4. The method of claim 2, further comprising:~~

~~receiving an update to the code image;~~

~~determining whether one code image is designated as non-operational; and~~

~~overwriting the code image designated as non-operational with the received update to the code image if one code image is designated as non-operational.~~

~~5. The method of claim 4, further comprising:~~

~~determining an earliest version of the code images in the memory device; and~~

overwriting the determined earliest version of the code image if one code image is not designated as non-operational.

Sub A5

6. The method of claim 4, further comprising:

determining whether one code image is corrupted; and

if one code image is corrupted, overwriting the corrupted code image with the received update before determining whether one code is non-operational.

7. The method of claim 1, wherein the first operation routine comprises a reboot routine and the second operation routine comprises an initialization routine.

8. The method of claim 7, further comprising:

incrementing the second counter if the initialization routine successfully completed;

rebooting if the initialization routine failed; and

performing another iteration of all previous steps after rebooting.

9. The method of claim 7, further comprising:

selecting one copy of the code image, wherein the executed initialization routine is a component of the selected code image, wherein the selected code image is designated as non-operational if the first counter is the first value and the second counter is the second value; and

selecting one other copy of the code image if the selected code image is designated as non-operational.

10. The method of claim 1, wherein the code image comprise different versions of the code image.

11. The method of claim 1, wherein the first operation routine comprises a reboot routine and the second operation routine comprises an initialization routine, and wherein the code images include a function routine to perform an operation after initialization, further comprising:

executing the function routine in one code image;

incrementing a third counter associated with the code image including the executed function routine if the function routine succeeded; and

using the third counter, in addition to the first and second counters, to select one of the multiple copies of the code image from the memory device to execute.

Sub A5

12. The method of claim 11, further comprising:

designating one code image as operational if the first, second, and third counters satisfy at least one threshold value, wherein the code image designated as operational is automatically selected from the memory device to execute after subsequent reboot operations.

13. The method of claim 11, further comprising:

designating one code image as non-operational if the first, second, and third counters satisfy at least one threshold value, wherein one other code image not designated as non-operational is selected from the memory device and executed.

14. The method of claim 13, wherein the threshold value for the third counter is zero and wherein the at least one threshold value for the first and second counters is greater than zero

15. The method of claim 13, further comprising:

incrementing the second counter if the initialization routine successfully completed;

~~rebooting if the initialization or function routine failed; and~~

~~performing another iteration of all previous steps after rebooting.~~

Sub A5

~~16. The method of claim 11, wherein the code image includes multiple function routines, wherein there is one counter for each of the multiple function routines, and further comprising:~~

~~designating one code image as operational if the first counter, second counter, and each counter associated with a function routine satisfy at least one threshold value, wherein the code image designated as operational is automatically selected from the memory device to execute after subsequent reboot operations.~~

~~17. The method of claim 1, wherein one operation routine comprises one of a reboot routine, an initialization routine or a function routine to perform a device specific operation.~~

~~18. A computer system for selecting a code image during a reboot routine, comprising:~~

~~a processor;~~

~~a memory device maintaining multiple code images, wherein the processor is capable of accessing them memory device;~~

~~program logic executed by the processor, wherein the program logic causes the processor to perform:~~

~~(i) executing a first operation routine;~~

~~(ii) incrementing a first counter if the first operation routine succeeds;~~

~~executing a second operation routine;~~

~~(iii) incrementing a second counter if the second operation routine succeeds; and~~

(iv) using the first and second counters to select one of the code images from the memory device to execute.

Sub A5

19. The system of claim 18, wherein the program logic is further capable of causing the processor to perform:

designating one code image as non-operational if the first counter is a first value and the second counter is a second value, wherein one other code image not designated as non-operational is selected to execute.

20. The system of claim 19, wherein the first value is greater than zero and the second value is zero.

21. The system of claim 19, wherein the program logic is further capable of causing the processor to perform:

receiving an update to the code image;

determining whether one code image is designated as non-operational; and

overwriting the code image designated as non-operational with the received update to the code image if one code image is designated as non-operational.

22. The system of claim 21, wherein the program logic is further capable of causing the processor to perform:

determining an earliest version of the code images in the memory device; and

overwriting the determined earliest version of the code image if one code image is not designated as non-operational.

23. The system of claim 21, wherein the program logic is further capable of causing the processor to perform:

determining whether one code image is corrupted; and if one code image is corrupted, overwriting the corrupted code image with the received update before determining whether one code is non-operational.

Sub A5

24. The system of claim 18, wherein the first operation routine comprises a reboot routine and the second operation routine comprises an initialization routine.

25. The system of claim 24, wherein the program logic is further capable of causing the processor to perform:

incrementing the second counter if the initialization routine successfully completed;

rebooting if the initialization routine failed; and

performing another iteration of all previous steps after rebooting.

26. The system of claim 24, wherein the program logic is further capable of causing the processor to perform:

selecting one copy of the code image, wherein the executed initialization routine is a component of the selected code image, wherein the selected code image is designated as non-operational if the first counter is the first value and the second counter is the second value; and

selecting one other copy of the code image if the selected code image is designated as non-operational.

27. The system of claim 18, wherein the code image comprise different versions of the code image.

28. The system of claim 18, wherein the first operation routine comprises a reboot routine and the second operation routine comprises an initialization routine, and wherein the code images include a function routine to perform an operation after

Sub A5

initialization, wherein the program logic is further capable of causing the processor to perform:

executing the function routine in one code image;

incrementing a third counter associated with the code image including the executed function routine if the function routine succeeded; and

using the third counter, in addition to the first and second counters, to select one of the multiple copies of the code image to from the memory device to execute.

29. The system of claim 28, wherein the program logic is further capable of causing the processor to perform:

designating one code image as operational if the first, second, and third counters satisfy at least one threshold value, wherein the code image designated as operational is automatically selected from the memory device to execute after subsequent reboot operations.

30. The system of claim 28, wherein the program logic is further capable of causing the processor to perform:

designating one code image as non-operational if the first, second, and third counters satisfy at least one threshold value, wherein one other code image not designated as non-operational is selected from the memory device and executed.

31. The system of claim 30, wherein the threshold value for the third counter is zero and wherein the at least one threshold value for the first and second counters is greater than zero

32. The system of claim 30, wherein the program logic is further capable of causing the processor to perform:

~~incrementing the second counter if the initialization routine successfully completed;~~

~~rebooting if the initialization or function routine failed; and~~

~~performing another iteration of all previous steps after rebooting.~~

Sub A5

~~33. The system of claim 28, wherein the code image includes multiple function routines, wherein there is one counter for each of the multiple function routines, and further comprising:~~

~~designating one code image as operational if the first counter, second counter, and each counter associated with a function routine satisfy at least one threshold value, wherein the code image designated as operational is automatically selected from the memory device to execute after subsequent reboot operations.~~

~~34. The system of claim 18, wherein one operation routine comprises one of a reboot routine, an initialization routine or a function routine to perform a device specific operation.~~

~~35. An article of manufacture for selecting a code image during a reboot routine, wherein the article of manufacture includes code in a computer readable medium capable of causing a processor to perform:~~

~~maintaining multiple code images;~~

~~executing a first operation routine;~~

~~incrementing a first counter if the first operation routine succeeds;~~

~~executing a second operation routine;~~

~~incrementing a second counter if the second operation routine succeeds; and~~

~~using the first and second counters to select one of the code images to execute.~~

Sub A5

36. The article of manufacture of claim 35, wherein the article of manufacture code is further capable of causing the processor to perform:

designating one code image as non-operational if the first counter is a first value and the second counter is a second value, wherein one other code image not designated as non-operational is selected to execute.

37. The article of manufacture of claim 36, wherein the first value is greater than zero and the second value is zero.

38. The article of manufacture of claim 36, wherein the article of manufacture code is further capable of causing the processor to perform:

receiving an update to the code image;

determining whether one code image is designated as non-operational; and

overwriting the code image designated as non-operational with the received update to the code image if one code image is designated as non-operational.

39. The article of manufacture of claim 38, wherein the article of manufacture code is further capable of causing the processor to perform:

determining an earliest version of the code images; and

overwriting the determined earliest version of the code image if one code image is not designated as non-operational.

40. The article of manufacture of claim 38, wherein the article of manufacture code is further capable of causing the processor to perform:

determining whether one code image is corrupted; and

if one code image is corrupted, overwriting the corrupted code image with the received update before determining whether one code is non-operational.

41. The article of manufacture of claim 35, wherein the first operation routine comprises a reboot routine and the second operation routine comprises an initialization routine.

Sub A5

42. The article of manufacture of claim 41, wherein the article of manufacture code is further capable of causing the processor to perform:

incrementing the second counter if the initialization routine successfully completed;

rebooting if the initialization routine failed; and

performing another iteration of all previous steps after rebooting.

43. The article of manufacture of claim 41, wherein the article of manufacture code is further capable of causing the processor to perform:

selecting one copy of the code image, wherein the executed initialization routine is a component of the selected code image, wherein the selected code image is designated as non-operational if the first counter is the first value and the second counter is the second value; and

selecting one other copy of the code image if the selected code image is designated as non-operational.

44. The article of manufacture of claim 35, wherein the code image comprise different versions of the code image.

45. The article of manufacture of claim 35, wherein the first operation routine comprises a reboot routine and the second operation routine comprises an initialization routine, and wherein the code images include a function routine to perform an operation after initialization, wherein the article of manufacture code is further capable of causing the processor to perform:

executing the function routine in one code image;

incrementing a third counter associated with the code image including the executed function routine if the function routine succeeded; and

using the third counter, in addition to the first and second counters, to select one of the multiple copies of the code image to execute.

Sub A5

46. The article of manufacture of claim 45, wherein the article of manufacture code is further capable of causing the processor to perform:

designating one code image as operational if the first, second, and third counters satisfy at least one threshold value, wherein the code image designated as operational is automatically selected to execute after subsequent reboot operations.

47. The article of manufacture of claim 45, wherein the article of manufacture code is further capable of causing the processor to perform:

designating one code image as non-operational if the first, second, and third counters satisfy at least one threshold value, wherein one other code image not designated as non-operational is selected and executed.

48. The article of manufacture of claim 47, wherein the threshold value for the third counter is zero and wherein the at least one threshold value for the first and second counters is greater than zero

49. The article of manufacture of claim 47, wherein the article of manufacture code is further capable of causing the processor to perform:

incrementing the second counter if the initialization routine successfully completed;

rebooting if the initialization or function routine failed; and

performing another iteration of all previous steps after rebooting.

Sub A5

50. The article of manufacture of claim 45, wherein the code image includes multiple function routines, wherein there is one counter for each of the multiple function routines, wherein the article of manufacture code is further capable of causing the processor to perform:

designating one code image as operational if the first counter, second counter, and each counter associated with a function routine satisfy at least one threshold value, wherein the code image designated as operational is automatically selected to execute after subsequent reboot operations.

51. The article of manufacture of claim 35, wherein one operation routine comprises one of a reboot routine, an initialization routine or a function routine to perform a device specific operation.

add A5